How do I create an empty array/matrix in NumPy?

Asked 11 years, 2 months ago Active 8 days ago Viewed 917k times

301

80

I can't figure out how to use an array or matrix in the way that I would normally use a list. I want to create an empty array (or matrix) and then add one column (or row) to it at a time.

At the moment the only way I can find to do this is like:

```
mat = None
for col in columns:
    if mat is None:
        mat = col
    else:
        mat = hstack((mat, col))
```

Whereas if it were a list, I'd do something like this:

```
list = []
for item in data:
    list.append(item)
```

Is there a way to use that kind of notation for NumPy arrays or matrices?

python arrays numpy

edited Dec 30 '14 at 18:33
MERose
2,717 ● 3 ● 33 ● 58

asked Feb 20 '09 at 9:58
Ben
54.2k ● 33 ● 79 ● 102

12 Answers

Active Oldest Votes

429

You have the wrong mental model for using NumPy efficiently. NumPy arrays are stored in contiguous blocks of memory. If you want to add rows or columns to an existing array, the entire array needs to be copied to a new block of memory, creating gaps for the new elements to be stored. This is very inefficient if done repeatedly to build an array.

In the case of adding rows, your best bet is to create an array that is as big as your data set will eventually be, and then add data to it row-by-row:

```
>>> import numpy
>>> a = numpy.zeros(shape=(5,2))
>>> a
array([[ 0.,  0.],
   [ 0.,  0.],
   [ 0.,  0.],
   [ 0.,  0.],
   [ 0.,  0.]])
>>> a[0] = [1,2]
>>> a[1] = [2,3]
>>> a
array([[ 1.,  2.],
   [ 2.,  3.],
   [ 0.,  0.],
   [ 0.,  0.],
   [ 0.,  0.]])
```

edited Apr 1 '10 at 12:10
Peter Mortensen
25.7k ● 21 ● 90 ● 118

answered Feb 20 '09 at 10:36
Stephen Simmons
6,076 ● 2 ● 17 ● 12

119 There is also numpy.empty() if you don't need to zero the array. – janneb Apr 19 '09 at 21:19

21 What's the benefit of using empty() over zeros()? – Zach Sep 1 '12 at 16:11

44 that if you're going to initialize it with your data straight away, you save the cost of zeroing it. – marcorossi Nov 13 '12 at 9:23

16 @maracorossi so `.empty()` means one can find random values in the cells, but the array is created quicker than e.g. with `.zeros()` ? – user3085931 Jul 13 '16 at 17:38

6 @user3085931 yep ! – Nathan Sep 30 '16 at 15:33

95

A NumPy array is a very different data structure from a list and is designed to be used in different ways. Your use of `hstack` is potentially very inefficient... every time you call it, all the data in the existing array is copied into a new one. (The `append` function will have the same issue.) If you want to build up your matrix one column at a time, you might be best off to keep it in a list until it is finished, and only then convert it into an array.

e.g.

```
mylist = []
for item in data:
    mylist.append(item)
mat = numpy.array(mylist)
```

`item` can be a list, an array or any iterable, as long as each `item` has the same number of elements.
In this particular case (`data` is some iterable holding the matrix columns) you can simply use

```
mat = numpy.array(data)
```

(Also note that using `list` as a variable name is probably not good practice since it masks the built-in type by that name, which can lead to bugs.)

EDIT:

If for some reason you really do want to create an empty array, you can just use `numpy.array([])`, but this is rarely useful!

edited Feb 8 '10 at 17:39 answered Feb 20 '09 at 10:44

Peter Mortensen 25.7k ● 21 ● 90 ● 118 Greg Ball 2,723 ● 17 ● 15

1 Are numpy arrays/matrices fundamentally different from Matlab ones? – levesque Nov 11 '10 at 3:20

1 If for some reason you need to define an empty array, but with fixed width (e.g. `np.concatenate()`), you can use: `np.empty((0, some_width))`. 0, so your first array won't be garbage. – NumesSanguis Sep 1 '17 at 5:56

54

To create an empty multidimensional array in NumPy (e.g. a 2D array `m*n` to store your matrix), in case you don't know `m` how many rows you will append and don't care about the computational cost Stephen Simmons mentioned (namely re-buildinging the array at each append), you can squeeze to 0 the dimension to which you want to append to: `X = np.empty(shape=[0, n])`.

This way you can use for example (here `m = 5` which we assume we didn't know when creating the empty matrix, and `n = 2`):

```
import numpy as np

n = 2
X = np.empty(shape=[0, n])

for i in range(5):
    for j  in range(2):
        X = np.append(X, [[i, j]], axis=0)

print X
```

which will give you:

```
[[ 0.  0.]
 [ 0.  1.]
 [ 1.  0.]
 [ 1.  1.]
 [ 2.  0.]
 [ 2.  1.]
 [ 3.  0.]
 [ 3.  1.]
 [ 4.  0.]
 [ 4.  1.]]
```

edited Apr 14 '17 at 2:46 answered Apr 10 '14 at 4:34

Franck Dernoncourt 53.5k ● 46 ● 254 ● 405

1 This should be the answer to the question OP asked, for the use case where you don't know #rows in advance, or want to handle the case that there are 0 rows – Spcogg the second Aug 15 '19 at 4:52

I looked into this a lot because I needed to use a numpy.array as a set in one of my school

26

projects and I needed to be initialized empty... I didn't found any relevant answer here on Stack Overflow, so I started doodling something.

```
# Initialize your variable as an empty list first
In [32]: x=[]
# and now cast it as a numpy ndarray
In [33]: x=np.array(x)
```

The result will be:

```
In [34]: x
Out[34]: array([], dtype=float64)
```

Therefore you can directly initialize an np array as follows:

```
In [36]: x= np.array([], dtype=np.float64)
```

I hope this helps.

edited Feb 7 '16 at 3:13 gsamaras 63.3k ● 30 ● 132 ● 224

answered Apr 10 '13 at 12:39 Andrei Paga 261 ● 3 ● 3

This does not work for arrays, as in the question, but it can be useful for vectors. – divenex Dec 22 '17 at 16:31

`a=np.array([])` seems to default to `float64` – P i Sep 7 '19 at 9:58

7

You can use the append function. For rows:

```
>>> from numpy import *
>>> a = array([10,20,30])
>>> append(a, [[1,2,3]], axis=0)
array([[10, 20, 30],
       [1, 2, 3]])
```

For columns:

```
>>> append(a, [[15],[15]], axis=1)
array([[10, 20, 30, 15],
       [1, 2, 3, 15]])
```

EDIT
Of course, as mentioned in other answers, unless you're doing some processing (ex. inversion) on the matrix/array EVERY time you append something to it, I would just create a list, append to it then convert it to an array.

edited Aug 30 '13 at 10:20 pradyunsg

answered Feb 20 '09 at 10:27 Il-Bhima

8,722 ●3 ●30 ●71   9,900 ●1 ●41 ●49

3

You can apply it to build any kind of array, like zeros:

```
a = range(5)
a = [i*0 for i in a]
print a
[0, 0, 0, 0, 0]
```

edited Oct 1 '15 at 18:07   answered Oct 1 '15 at 17:50
Ali G
29 ●2

4 If you want to do that in pure python, `a= [0] * 5` is the simple solution – Makers_F Dec 22 '15 at 3:46

3

Here is some workaround to make numpys look more like Lists

```
np_arr = np.array([])
np_arr = np.append(np_arr , 2)
np_arr = np.append(np_arr , 24)
print(np_arr)
```

OUTPUT: array([2., 24.])

answered Feb 5 at 8:41
Darius
337 ●2 ●16

3

If you absolutely don't know the final size of the array, you can increment the size of the array like this:

```
my_arr = numpy.zeros((0,5))
for i in range(3):
    my_arr=numpy.concatenate( ( my_arr, numpy.ones((1,5)) ) )
print(my_arr)

[[ 1.  1.  1.  1.  1.]  [ 1.  1.  1.  1.  1.]  [ 1.  1.  1.  1.  1.]]
```

- Notice the `0` in the first line.
- `numpy.append` is another option. It calls `numpy.concatenate`.

answered Sep 6 '11 at 21:20
cyborg
9,117 ●4 ●31 ●54

Depending on what you are using this for, you may need to specify the data type (see 'dtype').

2

For example, to create a 2D array of 8-bit values (suitable for use as a monochrome image):

```
myarray = numpy.empty(shape=(H,W),dtype='u1')
```

For an RGB image, include the number of color channels in the shape: `shape=(H,W,3)`

You may also want to consider zero-initializing with `numpy.zeros` instead of using `numpy.empty`. See the note here.

answered Sep 11 '16 at 0:28
nobar
34.2k ● 11 ● 106 ● 121

1

I think you can create empty numpy array like:

```
>>> import numpy as np
>>> empty_array= np.zeros(0)
>>> empty_array
array([], dtype=float64)
>>> empty_array.shape
(0,)
```

This format is useful when you want to append numpy array in the loop.

answered Aug 30 '19 at 13:47
veeresh d
29 ● 4

1

I think you want to handle most of the work with lists then use the result as a matrix. Maybe this is a way ;

```
ur_list = []
for col in columns:
    ur_list.append(list(col))

mat = np.matrix(ur_list)
```

answered Oct 9 '18 at 6:43
runo
21 ● 7

-1

For creating an empty NumPy array without defining its shape there is to way:

1.

```
arr = np.array([])
```

preferred. cause you know you will be using this as numpy.

2.

```
arr = []
# and use it as numpy. append to it or etc..
```

NumPy converts this to np.ndarray type afterward, without extra `[]` `dimionsion`.

answered Apr 14 at 6:23

Pedram

119 ● 11